



Department of Commerce Intranet Architecture Research Paper

Prepared for:

Dale Lanser, COR

U.S. Department of Commerce

OCIO, Office of Information Systems

Prepared by:

INDUS Corporation

1953 Gallows Road

Vienna, Virginia 22182

DOC Contract Number: 50CMAA900048

Task Order Number: ISE00032

Task 1: Standards and Protocols Research

October 10, 2000

<u>TABLE OF CONTENTS</u>	I
<u>INTRODUCTION</u>	1
<u>OPTION 1: BEST OF BREED COMMERCIAL SOLUTION</u>	2
<u>Overview</u>	2
<u>Personal Intranet Home Page Application</u>	2
<u>Single Sign-On</u>	2
<u>Authentication</u>	2
<u>Directory Application</u>	3
<u>System Cost</u>	3
<u>OPTION 2: OPEN SOURCE SOLUTION</u>	4
<u>Overview</u>	4
<u>Authentication</u>	4
<u>Directory Service / Directory Application</u>	5
<u>Portal Service</u>	6
<u>RECOMMENDATIONS</u>	7
<u>APPENDIX A: WEB PORTAL COMPONENTS</u>	9
<u>OVERVIEW OF PORTAL SYNDICATION FORMATS</u>	9
<u>References</u>	9
<u>IPLANET PORTAL SERVER</u>	11
<u>Architecture and Deployment</u>	11
<u>Authentication</u>	11
<u>Advantages</u>	12
<u>Disadvantages</u>	12

<u>References</u>	12
<u>APACHE JETSPPEED PORTAL</u>	14
<u>Overview</u>	14
<u>Advantages of Jetspeed</u>	14
<u>Disadvantages of Jetspeed</u>	15
<u>Conclusion</u>	15
<u>References</u>	15
<u>ENTRUST GETACCESS PORTAL</u>	17
<u>References</u>	17
<u>OTHERS</u>	18
<u>Oracle e Business Portal</u>	18
<u>Plumtree Corporate Portal 4.0</u>	18
<u>Open Source Portal Products</u>	18
<u>References</u>	18
<u>APPENDIX B: DIRECTORY SERVICE COMPONENTS</u>	20
<u>IPLANET DIRECTORY SERVER</u>	20
<u>Advantages</u>	20
<u>Disadvantages</u>	20
<u>References</u>	21
<u>NOVELL E DIRECTORY</u>	22
<u>Advantages</u>	22
<u>Disadvantages</u>	22
<u>References</u>	22
<u>OPENLDAP</u>	23
<u>References</u>	23

<u>MICROSOFT ACTIVE DIRECTORY</u>	24
<u>References</u>	24
<u>OTHERS</u>	25
<u>Innosoft IDDS</u>	25
<u>Critical Path inJoin</u>	25
<u>References</u>	25
<u>APPENDIX C: AUTHENTICATION COMPONENTS</u>	26
<u>TECHNICAL REVIEW OF WAP 0.3</u>	26
<u>Strengths of WAP 0.3</u>	26
<u>Potential Problems with WAP 0.3</u>	26
<u>Suggested Changes to WAP 0.3</u>	28
<u>MIT KERBEROS VERSION 5</u>	29
<u>Authentication and Single Sign-on Using Kerberos</u>	29
<u>Fundamentals of Kerberos Operations</u>	29
<u>Advantages of Kerberos</u>	30
<u>Disadvantages of Kerberos</u>	31
<u>Kerberos and the Web</u>	31
<u>Conclusion</u>	32
<u>References</u>	32
<u>PKI SOLUTIONS</u>	33
<u>References</u>	33

Introduction

This report is a compilation of data, references and recommendations relevant to the development of the Department of Commerce's intranet architecture. Information in this report has been compiled primarily from sources available on the Internet, and information obtained from direct interactions with product vendors.

We present two options as the best available technologies for this project. The first consists entirely of commercial, off the shelf components from iPlanet. iPlanet is a joint venture between Sun Microsystems and Netscape Communications, providing comprehensive solutions composed of products from each company's product lines.

The second recommended option is to integrate mostly open source software, primarily from the Apache Project. The open source components include the Apache web server, the Tomcat servlet engine, and the Jetspeed portal application. Novell's commercial Directory Server software (NDS) is required and the Web-user Authentication Protocol needs to be implemented to provide authentication services. Also, a substantial amount of custom software must be written to glue the applications together.

Both alternatives conform to industry standards for interoperability, making extensive use of technologies such as LDAP and XML. Importantly, both allow applications written in differing languages, such as Java, Perl and PHP, to be integrated into the architecture. Both provide a secure, extensible foundation for the Department's intranet.

The costs and life cycles of the two alternatives differ greatly, and these should be considered when making this decision.

The body of this document contains sections detailing the two options we recommend, and a *Recommendations* section detailing the strengths and weaknesses of the contenders.

The document concludes with a series of appendices covering the products and protocols researched in more technical detail: Appendix A, portals; Appendix B, directory services; and Appendix C, authentication protocols. The appendices cover the recommended solutions in detail and brief descriptions of other products that were evaluated.

iPlanet Portal Server

Overview

iPlanet Portal Server is the portal product of the Sun/Netscape alliance. In addition to portal page creation, it provides APIs for session management, access control, profile management, and logging. As such, it is a robust and nearly complete solution for most of the architecture required for the intranet infrastructure.

Netscape and Sun have provided scalable, robust enterprise infrastructures for years. It comes as no surprise that much of the industry considers the iPlanet offering the natural and unchallenged choice for such a system. On closer inspection, the product emerged as the clear leader for a commercial implementation to suit the Department's needs.

Below is a high-level summary that explains how the iPlanet Portal Server addresses key system needs. Additional technical details are included in Appendix A.

Personal Intranet Home Page Application

The portal delivery system is elegant. An end user can arrange and select display parameters for subscribed information and choose which optional subscription services they receive in an intuitive manner. It is a straightforward process for online application developers to provide portlet information summaries and launch links.

Single Sign-On

The authentication manager provides session information to online applications, so the end user does not need to log in again to any application that is accessed via the Portal Server. The session manager allows a configurable timeout, so users who do not remember to sign out of the intranet portal system present a limited risk to security.

Authentication

The Portal Server can use several means of authentication. For the Department intranet, LDAP authentication would be the primary means used. Clients connect via SSL to a login page, enter their username and password, which is passed from the Portal Server's authentication module to an external secure directory

server. The directory server indicates to the authentication module whether the combination passes or fails, which responds appropriately to the user.

Directory Application

The same external secure directory server used for LDAP authentication would be used as a back end for the Department's Directory Application. Neither the external directory server nor the Directory Application are part of the Portal Server, so the directory server would be purchased and the Directory Application would be written as part of this project.

Appendix B, Directory Server Components, discusses the candidate products. iPlanet Directory Server and Novell eDirectory are the leading choices. The cost of both products is the same and performance and administrative value for both are comparable.

For use in the commercial implementation, we recommend purchase of the iPlanet Directory Server. The Directory Server is used by the profile server within the Portal Server. It is likely the easiest to interface as an external directory server. It is available from the same source as the iPlanet Portal Server, so support would be from a single source.

System Cost

Because the Portal Server runs only on Sun Sparc/Solaris, a suitable host to run the Portal Server would have to be identified.

The Portal Server cost is \$90,000 per processor. It is likely that a strong single-processor host would support the Department intranet needs. For higher reliability and performance, the system could include two single-processor hosts with Portal Server installed.

The iPlanet Directory Server cost is \$2 per user. For use of the system by 35,000 users, the Directory Server cost is \$70,000. The iPlanet Directory Server runs on Red Hat Linux 6.0, so it can be supported by a relatively low-cost server.

Overview

A review of the open source components available leads to the conclusion that this effort can be done with mostly open source components. This approach requires a significant amount of glue to be written to bind together the various components. Additionally, the authentication service would need to be implemented, as no acceptable open source web authentication products exist that meet the Department's needs.

A point should be made here about what we mean when we talk about an open source solution. For the purposes of this document, we are referring to applications that are available in source code form that can be freely modified and distributed. This is not to be confused with free software. In fact, in most cases the open source software we recommend in this section is available both for free download and commercially from a vendor. The commercial distributions of these open source components typically include support, and we strongly recommend that support be purchased for all software components obtained for this effort.

Our recommended open source implementation combines Apache Jetspeed for the Portal Server, a yet to be developed WAP implementation to handle authentication, and a commercial directory server (either iPlanet Directory Server or Novell eDirectory). A Directory Application would need to be developed on top of the directory server.

Authentication

The centerpiece of the intranet architecture is the authentication services. The litmus test for product applicability turns out to be the single sign-on capability. Many authentication schemes are available, but there is no solid open source product in common use that provides single sign-on for a web environment.

We have reviewed the Web-user Authentication Protocol as submitted by Dennis Sutch, and found it to be a workable solution. We have concerns about the scalability of the protocol as currently defined, and have submitted some suggestions that would enhance scalability, at the cost of slower dissemination of user status changes to participating applications.

These changes are inspired by the mechanisms used in the Kerberos authentication protocol. Kerberos is a decade-old general-purpose authentication scheme that, unfortunately, is not directly applicable to our web authentication problem. However, the Kerberos developers have handled some

of the authentication issues we face, and some aspects of their solutions are applicable to this situation.

The Kerberos derived changes would alter the protocol so that instead of authenticating every hit to a WAP enabled application with both the WAP server and the Directory Server, users are granted time limited permissions to use applications. We recommend 10 minutes as a good default. After 10 minutes, the WAP and Directory Servers will be consulted again, and a new lease issued to the user.

To the users, this would appear no different than the current WAP specification, except that they would have an idle timeout mechanism. When a user stops using his browser with a WAP application for a period of time (10 minutes in this case), he will automatically be logged out of the system.

Unfortunately, the WAP server currently does not exist, and would have to be developed as part of this effort.

Additionally, the security model needs a few rounds of review and revision to ensure that there are no holes in it. Usually, when a new security protocol is released, it is rapidly updated by a series of patches to cover newly discovered design and implementation flaws. It generally takes a few iterations before new security protocols begin to be truly trustworthy.

With those caveats, the WAP protocol seems generally solid. The implementation should not be overly difficult, and the design is such that developers of WAP enabled Intranet Application will find it easy to integrate with the WAP architecture.

Directory Service / Directory Application

OpenLDAP, the open source directory server, is not yet ready for use in the Department intranet system. If the open source solution is selected, it will be necessary to purchase a commercial directory server for use as WAP and Directory Application data repository.

Appendix B, Directory Server Components, discusses the candidate products. iPlanet Directory Server and Novell eDirectory are the leading choices. The cost of both products is the same and performance and administrative value for both is comparable.

For use in the open-source implementation, we recommend purchase of Novell eDirectory. Novell's superior support for Linux, particularly Red Hat, makes it more in keeping with open-source philosophy. They are positioning themselves to keep pace with changes with the Linux community, which may prove to be of some benefit in the future.

Portal Service

Apache Jetspeed is a very good, open source Portal application. The Jetspeed architecture is outstanding. It is built on top of other Apache projects such as Cocoon, which transforms XML documents into other formats such as HTML, PDF and SGML, and Turbine, which is a complete document publishing framework.

One of Jetspeed's strengths is that it is completely configured via XML, and it communicates with content source providers using XML over HTTP. It speaks many of the XML based content syndication protocols, including RSS and OSC. And it also has a portlet architecture and API that can provide the basis for the inter-application communication functionality required by the Department.

The extensive use of XML makes Jetspeed very easy to integrate with the other components of this architecture.

Jetspeed has two sources of commercial support available, one from JavaCorporation via its e-Portal rebranding of Jetspeed, and the other from Xo3.com, via their application server OpenJODA. OpenJODA provides additional functionality, incorporating Jetspeed as a portal, JBoss as the EJB container, and Apache Tomcat as the servlet engine. OpenJODA is slated for its first non-beta release this month.

Given the rapidly evolving nature of these projects, we strongly recommend that any use of Jetspeed be backed by one of the above support options.

Recommendations

In researching the available products and protocols available for use in the Department's Intranet, we have found two very strong alternative approaches. Both the commercial and the open-source solutions will serve the needs of the Department well from a technical point of view. While the architectural differences are considerable, they will function in a similar manner with respect to the system users and developers. The biggest difference in the two solutions emerges when the entire life cycle of the product is examined.

The Sun/Netscape iPlanet alliance has an excellent COTS solution that provides a single source for portal, authentication and directory services. Although not an open source solution, it does adhere to open standards, and allows developers to freely choose the languages and environments in which to develop their intranet applications.

The COTS solution has the added benefit of support services directly from iPlanet, including both problem resolution and product upgrades. With the rapidly evolving nature of web technologies, it is important to have an Intranet platform that is adaptable to new paradigms and technologies as they evolve. For the relatively low price of annual maintenance, the Department can stay on top of these new opportunities through iPlanet upgrades.

Also important is the fact that the iPlanet solution is single source. Integrating products that are selected piecemeal is quite often very difficult, and in these cases the individual vendors are often non-responsive concerning interoperability problems. The single source COTS solution of the quality offered by iPlanet is both technically and economically sound.

The second option is to integrate products from the Apache Group (Tomcat, Jetspeed, etc.) with Novell eDirectory and the yet to be developed WAP server. With this option, all products except the NDS are open source.

NDS is included with this option instead of the more obvious choice of OpenLDAP simply because OpenLDAP is not yet up to task of managing large quantities of data in an enterprise environment. The directory services component is important enough to sacrifice open source for robustness and enhanced functionality.

The open source solution's greatest strength is that it provides near complete control to the Department. All the components can be freely modified to work as desired, and new functionality can be added as required.

The bulk of the cost of the open solution is the development of the WAP component and the integration of the various components to make them work

together. The open source nature of most of the components makes this possible. It would be much harder to try to integrate non-open source components. Also, there is a degree of risk that must be factored into any development effort, where purchase of a commercial product has a lower degree of risk.

A downside to this ease of integration is that custom changes made to the products must be later integrated with new product releases in order to take advantage of the new features and bug fixes. For example, if Jetspeed were modified to support integration with the intranet, those changes would need to propagate forward to future releases.

Like the COTS solution, the open source solution allows developers of intranet application to use languages and environments of their choosing to implement their applications. The open source solutions have an adherence to and utilization of open standards even greater than the COTS solution does. For example, Jetspeed makes extensive use of XML and XSLT for configuration, communication and processing.

So the two solutions are on par technically. The difference is primarily that of life cycle considerations. The COTS solution has relatively high up front costs in licensing, but upgrades and ongoing technical support are low. The open solution has high up front development costs, relatively low up front licensing costs, and ongoing code maintenance and upgrade costs.

Our recommendation is to implement the COTS solution provided by iPlanet. The advantages of single source software, low maintenance costs, and low development risks overcome the relatively high purchase costs. And although iPlanet is a closed source product, it embraces open standards and allows application development to occur on the platforms and languages preferred by the application developers.

If the initial cost of the COTS solution is prohibitive, our secondary recommendation is to integrate a commercially supported bundle of Jetspeed with Novell eDirectory and a custom developed WAP server. This provides a solid architectural basis and a large degree of customization options with low licensing costs. The implementation risks and long-term maintenance cost are higher, but the resulting implementation is likely to result in comparable success.

OVERVIEW OF PORTAL SYNDICATION FORMATS

Most portals get at least some of their content from remote sites, such as a summary of news items on the popular Slashdot site. This is accomplished by the remote site (Slashdot in this instance) publishing a summary of items in a file accessible through their web server. There are a number of formats for these content syndication files. A nice summary of the available formats is listed in the reference table under Moreover.com.

The following content syndication formats are all XML based:

- Netscape RSS – Used on the popular MyNetscape.com site.
- Internet Alchemy Open Content Syndication (OCS)- Powerful and flexible.
- Microsoft Channel Definition Format (CDF)- Used for Microsoft Active Channels in Internet Explorer.
- Web Distributed Data Exchange (WDDX) – Backed by Allaire, nice APIs in multiple languages.
- Microsoft Digital Dashboard – Microsoft backed, comprehensive SDKs available for Microsoft platforms.

There are no obvious advantages to one of these formats over the others. Some are backed by major corporations, and have freely available SDKs and APIs to allow easy development of these documents. The various formats generally contain the same information in differing XML forms, and as such are easily translated among the various formats. There are many sites, such as moreover.com, which as a service provide a huge selection of such feeds, all translated into a multitude of formats.

The important consideration relative to the Department's requirements is that the portal solution implemented should support one or more of these formats.

References

Netscape Rich Site Summary	http://my.netscape.com/publish/help/mnn20/quickstart.html
----------------------------	---

Open Content Syndication	http://internetalchemy.org/ocs/index.phtml
Moreover.com	http://w.moreover.com/categories/
WDDX	http://www.wddx.org/
Microsoft Digital Dashboard	http://www.microsoft.com/solutions/km/DigitalDashboard.htm
Microsoft Channel Definition Format	http://msdn.microsoft.com/workshop/delivery/channel/cdf1/cdf1.asp

IPLANET PORTAL SERVER

iPlanet Portal Server is the portal product of the Sun/Netscape alliance. It is described functionally in the above section, *Option 1: Best of Breed Commercial Solution*.

Architecture and Deployment

The complete Portal Server system is composed of a gateway component, a profile server using iPlanet Directory Server, and the portal server, which contains authentication server, profile server, and other sub-components of the system. The Portal Server uses the iPlanet Web Server.

It is possible to configure multiple gateways and use DNS round-robin or a sophisticated third-party load-balancing system to distribute loading. For increased reliability, it is possible to run the portal server on multiple servers. For such a configuration, there would be only one profile server, and all portal servers would reference it.

In addition to handling portal page creation, it provides APIs for session management, access control, profile management, and logging that can be used in development of online applications.

The system uses a hierarchical role-based architecture. Each authenticated user in the system is assigned a unique role. That role determines the plug-able content, layout of the content, the look and feel, access control, and application parameters.

Roles can be segmented by creation of domains at the top of the hierarchy. There is an administrator role per domain, as well as an administrator role across all domains.

Once the user is authenticated by the system, they have established a virtual private connection to the intranet providing secure access to their portal information and online applications.

Authentication

The Portal Server can use several means of authentication. At the least expensive side of the security spectrum is LDAP authentication. Users submit a username and password via SSL and the authentication module checks it against an LDAP directory entry. If it matches, the user has gained access to their customized home page, which contains links to their online applications.

A free but administratively costly way of increasing security utilizes Bellcore's S/Key system. The S/Key generator is software that runs on the Solaris machine that hosts the Portal Server. Employees who use S/Key must log into the Solaris host as themselves and run the S/Key generator program to obtain a list of

single-use pass-phrases. Each will succeed only once for entry into the system. The administrative cost of user Solaris accounts and complexity of the operation for non-technical employees render this practically unusable for the Department's purposes.

The system is capable of using RSA SecurID for authentication. SecurID is token-based. The user must carry a smart-card or similar token-bearing device and can only log in using a device that is able to read the token and present it over the network to the authentication module. As such, it is unsuitable for use by the Department for this system.

The system can use SafeWord, which recently merged with Secure Computing Corporation. The SafeWord user carries a small hand-held device that generates single-use passwords. It would be too expensive financially and administratively to deploy at large. Should the Department choose the iPlanet Portal Server to deploy the intranet, it might be worth considering this as a means of further securing priority connections such as remote administrative operations in the future.

Advantages

The Portal Server solves many of the infrastructure challenges posed by construction of the Department intranet without the need for additional code to be written. The vendor has many years experience solving all aspects of the problems involved, so the product is mature, stable, and robust.

Disadvantages

The primary disadvantage is cost. The version of the iPlanet Portal Server that runs on SSL costs \$90,000 per processor, and runs only on Sun Sparc/Solaris platform. The Directory Server for authentication and the Directory Application would be purchased separately and costs \$2 per user.

A secondary disadvantage is that the solution is not open-source. This would be of more concern except that iPlanet has attempted to use open protocols and standards within this product and their general offerings wherever possible.

References

iPlanet Home	http://www.iplanet.com/
iPlanet Portal Server Home	http://www.iplanet.com/products/infrastructure/portal/index.html
RADIUS Info	http://www.infoseceng.com/radius.htm
SafeWord Home	http://www.safeword.com/welcome.htm

SecurID Info	http://www.rsasecurity.com/products/securid/
ZDNet Portal Evaluation	http://www.zdnet.com/filters/printerfriendly/0,6061,2565900-54,00.html

APACHE JETSPPEED PORTAL

Overview

Apache Jetspeed is an open source portal development project, managed under the Apache project. It is based on Java servlets, and uses XML/XSL extensively. As such, it requires a web server and servlet engine to properly function. We recommend Apache web server and Apache Tomcat servlet engine for these roles. Alternately, Apache JServ could replace Tomcat. JServ is the older servlet engine, while Tomcat is the newer, still somewhat developmental version. Since Jetspeed is currently developed under Tomcat, and Tomcat supports the newer servlet API specification, we recommend it over JServ.

Recently, JavaCorporate (Jcorporate Ltd.) has started providing a download/support bundle of Jetspeed named JavaCorporate ePortal. ePortal is priced at \$599 per year.

Also, Swiss based Xo3.com has announced OpenJODA, an enterprise class application server based in part on Jetspeed. OpenJODA consists of Apache Tomcat as the servlet engine, Jetspeed as the enterprise portal, and JBoss as the Enterprise Java Bean (EJB) container. The product is scheduled for a fall rollout, but has a currently available prerelease version. Pricing is not yet available.

Jetspeed is built on a foundation of Java 1.1.8 or higher, Java Servlet Development Kit 2.0 (JSDK2.0), and Cocoon, a sibling apache project which provides XML/XSL publishing framework. As such, it is completely open source, and platform independent. It will theoretically run on any JDK/JSDK compliant platform, but has only been tested under Sun and IBM virtual machines, and Apache Jserv and Apache Jakarta Tomcat Servlet engines.

Jetspeed accepts content syndication feeds from a number of sources, including XML based systems such as Rich Site Summary (RSS), SMTP feeds, and newer protocols like iCalendar. Jetspeed features a Portlet API so content can be plugable. It also supports Avantgo for PalmOS users.

Advantages of Jetspeed

Jetspeed is open source and low cost.

The support provided by JavaCorporate and Xo3.com makes Jetspeed much more attractive as a enterprise level solution. Installation of Jetspeed is nontrivial, though configuration is nearly completely web-based once the initial installation is completed. Given the youth of Jetspeed, having access to support staff familiar with the vagaries of the individual releases is invaluable.

Jetspeed is configured entirely in XML, which makes interoperability with other systems easy. It supports multiple schemas within XML, including RSS, the most widely used content syndication protocol.

Jetspeed provides a Portlet API. Although more thorough research is required, we believe that this API might provide the basis for the inter-application communication protocol required in this project.

Disadvantages of Jetspeed

Jetspeed is as yet an immature product. There are no big sites running Jetspeed in a production environment exposed to the public Internet. We have no way of knowing the extent of Jetspeed's use in private Intranets. Reviewing the architecture of Jetspeed, we believe that it is architecturally sound, and that the upcoming 1.2 release will be followed by wide adoption.

One symptom of being an immature product is that the developer base is relatively small at this time. All the developers are actively working on functionality, leaving the documentation and installation tasks lacking. The corporate support provided by JavaCorporation and Xo3.com should overcome this shortcoming.

Conclusion

While Jetspeed must be considered a young application at this time, it is far better architected than any of the other open source portal applications we have investigated. We believe that it will quickly become widespread, and that it will be easy to adapt Jetspeed to the Department's requirements.

References

Jetspeed Overview	http://java.apache.org/jetspeed/
XML.com's Jetspeed Review	http://www.xml.com/pub/2000/05/15/jetspeed/
Java Corporate	http://www.javacorporate.com/
The Apache Project	http://www.apache.org/
The Cocoon Project	http://xml.apache.org/cocoon/
Apache XML Project	http://xml.apache.org/
An Interview with the Developer of Jetspeed	http://www.devshed.com/Server_Side/JServ/JetSpeed_Interview/

Jserv Project	http://java.apache.org/jserv/
Tomcat Project	http://jakarta.apache.org/tomcat/index.html
Jboss	http://www.jboss.org/

ENTRUST GETACCESS PORTAL

Entrust recently acquired enCommerce's getAccess portal product (June 2000). getAccess works in conjunction with the Netscape/iPlanet Directory Server product to provide many of the features required by the DOC Intranet (single sign-on, user authentication). It includes support for session management, rule and role-based access control, delegable administrative features, and sophisticated logging. The administration of getAccess is completely web-based using Java applets. Although an Entrust product, getAccess has not yet been integrated with Entrust's PKI products at all.

A getAccess server costs \$10,000 per installation plus \$6 per user (with a user base of 25,000 to 100,000). For DOC, this amounts to around \$220,000 for a single server hosting all 35,000 users making getAccess rather costly.

Furthermore, getAccess is not even a true web portal. Rather than merging content from various web applications onto a single web page, it creates a configurable menu of URLs to the desired web applications.

These facts make getAccess unsuitable for use with the DOC Intranet.

References

Entrust Home	http://www.entrust.com/
Entrust getAccess Portal Product	http://www.encommerce.com/

OTHERS

Oracle e Business Portal

The Oracle E-Business Portal is not suited to the needs of the DOC intranet. While it handles creation and modification of the customizable portal pages very well, it does not solve other problems involved in the project.

Plumtree Corporate Portal 4.0

Plumtree is a popular, Microsoft Windows only corporate portal that is quite mature. It is, however, extremely proprietary.

Instead of an open API for portlets, there are “Plumtree Portal Gadgets”. There is a proprietary “Plumtree Portal Transformer” for media conversion and content feeds.

They do provide a built in search and indexing engine, and it is possible to use a standard LDAP server for authentication chores.

There is work underway to provide single sign-on capability, through a partnership with Netegrity. This is to be accomplished by integrating Corporate Portal with the Netegrity Siteminder platform.

Overall, Corporate portal 4.0 is too heavily Windows dependent, too proprietary, and does not yet boast single sign on capabilities.

Open Source Portal Products

There are a number of open source portal products that we examined. With the exception of Jetspeed, most turned out to be poorly architected scripts, or products in the very earliest stages of development.

For completeness, we have listed some of the open source portals we examined in the references.

References

Oracle E-Business Portal	http://www.oracle.com/portals/index.html?intro.html
Plumtree Corporate Portal 4.0	http://www.plumtree.com/product/default.htm
Community Portal (CommPort)	http://www.tc.ca/commport/

The Linux Portaloo v0.2	http://www.linux.org.uk/cgi-bin/portaloo
lloop: Open Source Software for Community Portals	http://www.lloop.com/html/docIndex.pyp
ODP++	http://www.portalscripts.com/

IPLANET DIRECTORY SERVER

The iPlanet Directory Server is said by iPlanet and third-party evaluators to possess 60-70% of the directory service market share. Third-party evaluations have found it to provide the best performance and reliability of the directory services currently available.

The Directory Server supports all LDAP search filters as well as approximate searches. It fully supports LDIF format and has a change log.

Administrative controls are excellent. It offers an intuitive Java console, but any operation that the console can perform is available in command-line form as well.

It is easily configured for replication. One server is a designated master and allows replication to and from slave servers. The model supports synchronizing by days and times or 'always keep in sync.'

iPlanet also offers a directory access router which provides automatic fail-over, load balancing, and additional security capabilities.

Also worthy of note is iPlanet's acquisition of Innosoft, makers of IDDS. IDDS rated a very close second LDAP directory server to iPlanet's Directory Server: in some ways trailing performance, but in others exceeding it. iPlanet intends to incorporate their expertise into the version 5.0 offering scheduled for late this year.

Advantages

It is consistently the highest-rated directory server. It is easy to use, well documented, and has by far the largest installed base. It has the best performance and highest reliability rating of available directory servers.

If the iPlanet Portal Server is selected, it will be available from the same source as the rest of the infrastructure and share administrative philosophy and support.

Disadvantages

Cost is a disadvantage. It and its closest competitor, Novell eDirectory, have the same cost- \$2 per user. Intraware, the iPlanet vendor we have worked with, made it clear that they were willing to negotiate that price for large numbers of users.

Novell's comparison literature observes that the iPlanet Directory Server is LDAP-only and that eDirectory is a full-service directory service with several

interfaces and a long list of vendors that provide companion products. While this may be a general disadvantage, it is not relevant to its intended use in the Department intranet system.

A slight disadvantage is that iPlanet is less devoted to its Linux deployment than Novell. iPlanet's current version supports Red Hat 6.0 but not 6.2 and they have no plan for support of 7.0, even in their next version.

References

iPlanet Home	http://www.iplanet.com/
IPlanet Directory Server	http://www.iplanet.com/products/infrastructure/dir_security/dir_srvr/
IPlanet Directory Access Router	http://www.iplanet.com/products/infrastructure/dir_security/idar/
Network World Directory Service Comparison	http://www.nwfusion.com/reviews/2000/0515rev2.html
Intraware Directory Service Comparison (pdf)	http://www.3isystems.com/extaccess/IntrawareDirSvrCompare.pdf
Novell Comparison	http://www.novell.com/competitive/nds/nds-ipplanet.html

NOVELL EDIRECTORY

Novell eDirectory has excellent performance and reliability. While third-party evaluations rate it a close second to the iPlanet Directory Server, Novell claims to have found it to have better performance.

eDirectory fully supports all LDAP search filters and has a bulkload utility to load users in LDIF format.

LDAP administration must be performed via a Java-based administrative interface called ConsoleOne. It is possible to perform other administrative tasks via NetWare Manager.

Advantages

Novell has demonstrated a commitment to Linux support. In fact, on September 27, Red Hat announced that it had selected Novell eDirectory to provide the directory services infrastructure for Red Hat Network worldwide. Novell has positioned itself to keep pace with changes in the Linux world.

Disadvantages

Cost is a disadvantage. It and iPlanet Directory Server have the same cost- \$2 per user.

It has been rated slightly below the iPlanet Directory Server in performance and administrative ease, but was not far behind.

References

Novell eDirectory Home	http://www.novell.com/products/nds/index.html
Novell eDirectory Details	http://www.novell.com/products/nds/details.html
Network World Directory Service Comparison	http://www.nwfusion.com/reviews/2000/0515rev2.html
Intraware Directory Service Comparison (pdf)	http://www.3isystems.com/extaccess/IntrawareDirSvrCompare.pdf
Novell Comparison	http://www.novell.com/competitive/nds/nds-iplanet.html
Novell/Red Hat Press Release	http://www.novell.com/news/press/archive/2000/09/pr00099.html

OPENLDAP

OpenLDAP is an open source project that may soon be a contender among directory service options. The current stable release is OpenLDAP version 1.2.1.1, which implements LDAP version 2. The latest release, which is not yet considered stable, is 2.0.4. It implements LDAP version 3.

It is currently a reasonable choice for small, light-duty implementations, since the stable release has proven reliable and it is free. However, until it stably implements LDAP version 3 and performance improves, it is questionable for enterprise use.

While open source projects rarely progress on a defined schedule, it is worthy of note that one of our vendor technical contacts told us that several key developers of the OpenLDAP team were recently hired by Microsoft and Netscape, and progress is expected to be much slower than hoped. We were not able to confirm this by any other source.

References

OpenLDAP Home	http://www.openldap.org/
Network World Directory Service Review	http://www.nwfusion.com/reviews/2000/0515rev2side.html
Sunworld LDAP Products Review	http://www.sunworld.com/sunworldonline/swol-09-1999/swol-09-ldap2.html

MICROSOFT ACTIVE DIRECTORY

Microsoft included Active Directory, an LDAP compliant directory server, in its release of Windows 2000. It is a relatively new product, and has received mixed reviews. While most operations compete respectably with the performance leaders, bulk loading and synchronization operations are still surprisingly poor.

Although the current implementation of Active Directory is 100% LDAP compliant, Microsoft has a documented history of using open standards for low cost or free components until they have achieved wide acceptance and then deviating from them in such a way as to dilute or destroy the open standard in favor of proprietary Microsoft standards, thereby establishing reliance on Microsoft products. While Active Directory would be a poor choice as a component of the Department's system for performance reasons, it is also a risky choice given the Department's wish to use open standards whenever available.

References

Active Directory Overview	http://www.microsoft.com/windows2000/guide/server/features/dirlist.asp
Information Week Report	http://www.informationweek.com/777/ad.htm
Network World Directory Service Review	http://www.nwfusion.com/reviews/2000/0515rev2side.html

OTHERS

Innosoft IDDS

Innosoft has been acquired by the Sun/Netscape alliance. IDDS, which rated a close second to the iPlanet Directory Server is no longer available. Their web page promises that technical expertise will be put to good use in the future iPlanet Directory Server iDS V5.0 that is tentatively scheduled for the end of this year.

Critical Path inJoin

Critical Path is an important entity in the directory market. Their meta-directory offering is among the best available. Their current directory server offering, inJoin Directory Server, is the newest version of their Global Directory Server product. Comparisons of the previous version of Global Directory Server with iPlanet and Novell products revealed significantly lower performance, rendering it an ineffective choice for the Department's authorization directory and Directory Application.

References

Innosoft Home	http://www.innosoft.com/
Innosoft Announcement	http://www.innosoft.com/directory_solutions/eval-interest.html
Critical Path Home	http://www.cp.com/
Critical Path eJoin Product Family Home	http://www.cp.net/products/injoin_index.html

TECHNICAL REVIEW OF WAP 0.3

Strengths of WAP 0.3

Positive, instantaneous control of user access

WAP 0.3 does an excellent job of providing instantaneous control of user access. Changes to the directory service database or the WAP database are instantly reflected in the users permissions.

Custom solution

As WAP is a custom software solution that will be developed and maintained at the Department, it will exactly meet the current requirements for the intranet. Furthermore, required modifications and enhancements can be implemented without waiting for an upgrade or new release.

Potential Problems with WAP 0.3

Single point of failure

WAP 0.3 as written does not provide for any means of redundancy. If there is a failure in the WAP server (hardware, network, DOS attack, etc.), then all WAP enabled applications on the network are down. The protocol should be enhanced to facilitate replication.

Excessive latency

WAP 0.3 as written will cause long latency times for users requesting pages from a WAP enabled application, even in the common case. The most common case will be a user requesting a page, already possessing a WAP-session and WAP-application cookie. (See WAP 0.3, page 24). In this case, the user's browser opens a connection to the application server. During the ensuing WAP authentication, this connection remains open. The latencies present in the application server to WAP server connection (steps 4 & 8) and the latencies present in the WAP server to directory server (steps 4 and 6) all add up, holding the browser to application server connection open.

Heavy loading of WAP server and directory server.

For every single page served from within the WAP enabled community, a TCP connection is opened from that application server to the WAP server. Also, a

TCP connection is opened between the WAP server and the directory service for each page requested.

This will present a very heavy loading to the WAP server. Consider the case of the Time and Attendance application, webTA. We expect webTA to generate between 10 and 20 pages a second on average, with peaks closer to 50 to 75 hits per second when fully deployed at Department of Commerce. The webTA application is designed to be distributed/replicated (multiple web servers, multiple servlet engines, distributed databases), and makes extensive use of connection pooling to achieve its throughput rates. WAP will need to service all these requests, as well as the requests for all the remaining applications on the intranet.

Lack of timeout mechanism

WAP 0.3 appears to lack a mechanism for timing out user sessions due to inactivity. Users can log into the WAP cluster, and remain logged in for extensive periods of time with no activity. This is easily fixed in the current architecture by having the WAP server check each requests timestamp against the previous timestamp, and reject requests that come after a long period of inactivity.

Untested security model

New security protocols typically expose bugs and exploits early in their deployment. As WAP has not yet been developed or deployed, it must be considered immature at this point.

Peer review can provide some minimal sanity checks, but there is no substitute for time and exposure.

Potential exploits

A malicious WAP enabled application could store user password with little chance of detection.

On receiving a page request for an unauthenticated user, the application could present a login page that mimics WAP's login page. On receiving the username/password info, the application could send back a 'Directory service timed out page'. Only if the user noticed the bad URLs would this be detected.

With some study of the protocol, it may even be possible for the application server to spoof the user's IP address, submit the login request on the user's behalf directly to WAP, set the cookie on the user's browser, and redirect the user back to itself. This requires further investigation.

Suggested Changes to WAP 0.3

Share secret keys between WAP and WAP-enabled applications

A critical bottleneck in the WAP architecture is caused by the need for the WAP application to validate WAP-session cookies for each page requested by a user (WAP 0.3 protocol, page 24, step 4). An alternate method, which is used extensively by the Kerberos protocol, is to share a secret key between WAP and each of the applications it serves. (Each application would have a unique key shared only between that application and WAP). When WAP creates a WAP-application key to send to the browser (WAP 0.3 protocol, page 23, step 9), WAP encodes the application session with the particular applications secret key.

Later, when the application wants to verify the application key, it uses its secret key to decode the application session. If it decodes successfully, the application knows that the session came from the WAP server, and that it was valid at the time the key was created.

Use shared secret keys to facilitate replication

The concept of secret keys shared between applications and the WAP server can be leveraged to provide an easy means of distributing load across servers. By sharing keys between applications and the pool of WAP servers, a user should be able to seamlessly authenticate with any of WAP servers.

Use creation timestamps and duration values in application sessions

When the WAP server creates an application session key, it can encode a timestamp and duration into the cookie. A typical duration could be 10 minutes. If the application receives the cookie within the duration, it accepts it as valid. If the cookie has expired based on the duration, the application server contacts the WAP server to get a new application session key.

This provides two benefits. First, network traffic is reduced between the application server and the WAP. Only one communication is needed for each duration period, rather than one each page served.

Second, it provides an automatic timeout for users through inactivity.

On the downside, the WAP database and the Directory Service database are only consulted once every 10 minutes. Changes to the data in either of these databases could take (at most) 10 minutes to propagate back to the actual users browser (via the application itself).

MIT KERBEROS VERSION 5

Authentication and Single Sign-on Using Kerberos

Kerberos is a security authentication system initially designed by MIT's project Athena in 1987. Kerberos provides mutual authentication between a network application (service) and a user. That is, Kerberos ensures that the user can be confident that he really is talking to the service he expects to, and that the service knows the user it is talking to. Kerberos also provides a mechanism for enforcing permissions for users to use services on the net.

Kerberos is a very complex and mature security infrastructure. See the references for more in depth coverage of how Kerberos works, and what the shortcomings and limitations of Kerberos are.

Kerberos is covered in some detail in this section because, although it was unsuitable for direct inclusion in the Department's architecture, it contains some novel approaches and techniques that are directly applicable to the WAP protocol. In particular, Kerberos uses ticket time-out periods, and shared secret keys between both users and the authentication server and applications and the authentication server.

Fundamentals of Kerberos Operations

Kerberos provides an Authentication Server (also known just as a Kerberos Server) that provides authentication and permissions information to users wanting to use a system (the client), and indirectly to applications (the application) deciding whether to provide services to the user. It does so through the issuance of Tickets.

In order to make all of this work, a number of secrets are shared among the three participants. First, the user and the server both know the user's password. Second, the server and the application both know the application's secret password. And finally, when the server creates a ticket, it also creates a 'session key' that is passed on to both the client and the application.

A ticket is an encrypted set of data that a client can present to the application when requesting services.

A Kerberos session works something like this:

1. A user sits down at his workstation, and logs into the Kerberos system (using a program named kinit). Kinit sends a request, containing only the user's username and workstation address to the Kerberos server. The server creates a 'ticket-granting-ticket', encodes it with the user's password, and sends it to the user. For future communications, the server also includes a new session key, shared by the Kerberos server and the

user. The user must encrypt all further communications with this server with this session key.

2. The user receives the encoded ticket-granting-ticket, and decodes it (using the users password), and stores it on his local disk.
3. The user now wants to run a service, for example mail. Kerberos uses the concept of single sign-on, so the user does not have to enter his password again. Instead, he sends a request to the Kerberos server to get a mail ticket, and includes his ticket-granting-ticket as proof that he has already authenticated with the Kerberos server. This is all encrypted using the session key from step 1.
4. The Kerberos server receives the request, and looks up the session key it had for this user earlier to decrypt the package. Once decrypted, the Kerberos server sees that it is a request to use mail. It looks up in its database that this user is indeed allowed to use that mail server. The Kerberos server creates a 'mail-granting-ticket' to give to the user. Inside this ticket, it creates a message for the user to forward to the mail server, and encrypts this message with the mail-servers secret key. The mail servers secret key is shared only by the Kerberos server and the mail server. The user will be unable to decrypt it, but can only forward it to the mail server.
5. The user receives the multiply encrypted key from the Kerberos server. He decrypts it using the session key from step 1, and finds an ticket encrypted with the mail servers key inside. The user forwards this to the mail server, with a request to use mail.
6. The mail server gets the request from the user. It uses its secret key (shared only with the Kerberos server) to decrypt the ticket, which states that the user is allowed to use mail. It provides mail access to the user.

Advantages of Kerberos

1. Passwords are NEVER sent over the network, neither in an encrypted nor unencrypted state. Instead, packages are created and encrypted with passwords shared between the sender and receiver.
2. Applications and users are mutually authenticated.
3. Kerberos supports single-sign-on. Users only need to type their passwords once, and they then have access to all their applications.
4. Remote login. Users can login to Kerberos from any machine on the network, simply by providing username@domain as their login name.
5. Kerberos is open source.

6. Kerberos can be downloaded and installed for free, or it can be purchased with support.
7. Kerberos implementations exist for many platforms and applications.
8. Kerberos is a well understood standard. Windows 2000 uses Kerberos authentication for all Microsoft domain logins.

Disadvantages of Kerberos.

1. Applications that wish to use Kerberos (client applications) must be 'Kerberized'. This consists of linking to a Kerberos library and making Kerberos calls for authentication.
2. Most web-browsers do not support Kerberos. One notable exception is some early variants of Mosaic.

Kerberos and the Web.

There are two approaches to using Kerberos authentication in a web-based environment. The approaches are based on the location where the tickets are actually stored.

One approach is to have the web server manage the Kerberos authentication for the clients (browser users). The web server presents a login page, and the users password is sent to the web server via HTTP Basic Authentication methods. The web server then contacts the Kerberos server, and authenticates for the client.

This is a particularly bad approach, for a number of reasons.

First, single sign-on is not possible. The authentication (and the corresponding ticket-granting-ticket) is stored on the web server. The user could be authenticated for services on that web server, but nowhere else.

Second, the web server cannot act as a trusted agent for the users passwords. A malicious web server/admin could grab the users passwords, and be authenticated as that user anywhere.

Finally, the web server would have to maintain a mapping of the Kerberos tickets it holds against the clients at the other end of the connection, probably by using cookies to maintain the identity of the browser. This is a problem-prone undertaking, given the stateless nature of HTTP connections.

The second approach to web-based Kerberos authentication is to have the browsers manage the Kerberos authentication themselves. Unfortunately, most browsers have not been Kerberized - they do not know how to talk to a Kerberos server. Additionally, the web server would need to be kerberized in order to understand and accept the Kerberos ticket as an authentication method.

This approach is possible using existing solutions. There is a mod_kerberos apache web server module that allows the web server to understand Kerberos tickets submitted as authentication. And there are plug-ins for popular browsers which, in effect, kerberize the browser.

But in a large-scale intranet, the installation and maintenance cost of managing browser plug-ins on a large scale is prohibitive.

One alternative would be to use a signed Java applet downloaded from the web server to handle the Kerberos authentication tasks. The applet would need to be signed in order for the browser Java sandbox to allow connections to the Kerberos server.

Conclusion

At first review, Kerberos appears to meet many of the requirements for the Departments Intranet Infrastructure. However, Kerberos in its current implementation is not web friendly.

Development of a Java based applet to handle the browser authentication tasks would seem to be a possible solution to Kerberos integration in a web environment.

References

Designing an Authentication System: a Dialogue in Four Scenes	http://web.mit.edu/kerberos/www/dialogue.html
Kerberos 5 HOWTO	http://www.dimensional.com/~bgiles/krb5.html
Kerberos: An Authentication Service for Computer Networks	http://www.isi.edu/gost/publications/kerberos-neuman-tso.html
Pass-Through Proxying as a Solution to the Off-Campus Web-Access Problem	http://www.stg.brown.edu/pub/proxydoc/Proxy.tr98.1.shtml
The Moron's Guide to Kerberos, Version 1.2.2	http://www.isi.edu/~brian/security/kerberos.html
JGSS Package	http://choices.cs.uiuc.edu/Security/JGSS/jgss.html
Security Unit SSO project	http://security.dstc.edu.au/projects/sso/

PKI SOLUTIONS

PKI (public key infrastructure) is a combination of encryption technologies and services that facilitates the protection of communication and business transactions on a network. PKI integrates digital certificates, public key cryptography, and certificate authorities into a network security architecture. A typical PKI solution allows users and applications to be authenticated, allows users to digitally sign documents, and provides a set of tools for the management of the resources involved (users, certificates, applications). Some PKI implementations provide a single sign-on capability (Entrust Direct), but this is not a required part of the PKI infrastructure.

The two leading PKI implementations on the market today are from Verisign Incorporated and Entrust Technologies. Both of these product suites are rather costly and neither one addresses the portal requirements of the DOC intranet project. Entrust has a portal-like product called getAccess; however, it is more of a configurable menu system than a true portal (see the Entrust getAccess section of this document).

If the DOC intranet were to expand to include some or all of the PKI functionality, there are two alternatives that work with our proposed solutions. iPlanet has a certificate management systems that implements the PKI functionality. It integrates tightly with the Netscape Directory Server product.

There is also an open source PKI implementation underway named OSCAR (Open Secure Certificate ARchitecture). It is a relatively new effort, having started in the beginning of 1999, but it appears to have most of the core PKI functionality included in it at this time. It has support for certificates (generation, signing, verification) and digital signatures and it interfaces with LDAP directories.

References

Entrust Technologies PKI products	http://www.entrust.com/products/pki/
iPlanet Certificate Management System	http://www.iplanet.com/products/infrastructure/dir_security/cert_sys/
Oscar: DSTC's PKI Project	http://oscar.dstc.qut.edu.au/
Verisign Inc. PKI Product	http://www.verisign.com/enterprise/